A Practical Leakage-Resilient Signature Scheme in the Generic Group Model*

David Galindo and Srinivas Vivek

University of Luxembourg
{david.galindo,srinivasvivek.venkatesh}@uni.lu

Abstract. We propose a leakage-resilient signature scheme in the continual leakage model that is based on a well-known identity-based encryption scheme by Boneh and Boyen (Eurocrypt 2004). The proposed signature scheme is the most efficient among the existing schemes that allow for continual leakage. Its efficiency is close to that of non leakage-resilient pairing-based signature schemes. It tolerates leakage of almost half of the bits of the secret key at every new signature invocation. We prove the security of the new scheme in the generic bilinear group model.

Keywords: leakage-resilient cryptography, digital signature, continual leakage, generic group model, efficiency.

1 Introduction

Side channel attacks are often effective in recovering the secret key of cryptosystems that are provably secure otherwise [16,17,7]. Typical examples of side channel attacks include analysis of running-time, power consumption, electromagnetic radiation leak, fault detection, to name just but a few. Countermeasures adopted in practice against side channel attacks are usually heuristic, aimed often at covering a restricted class of attacks. On the other hand, it is desirable to extend the traditional provable security methodology to also include side channel attacks. This area of contemporary cryptography is usually referred to as *leakage-resilient cryptography* and it has been an increasingly active area in recent years.

In this work we make two main assumptions to model leakage:

- **Bounded leakage:** the useful leakage data per signature invocation is bounded in length (but unbounded overall);
- **Independent leakage:** the computation can be divided into rounds, where each such round leaks independently.

* Supported by the National Research Fund, Luxembourg C09/IS/04.

This model has been previously used in [11,21,15,12]. The first assumption can be seen overly restrictive; however it should be noticed that in practice many side-channel attacks only exploit a polylogarithmic amount of information. The second assumption allows us to divide the memory of a device, at every computing step, into two parts - an *active* and a *passive* part. The part of the memory being currently accessed by a computation is the active part, and only the active part leaks information at any given time. We stress that even if our leakage definition is local with respect to each part of the memory, it still captures some global functions of the secret key, for instance any affine leakage function. We refer to the work by Dziembowski and Faust [10] for a discussion on the significance and limitations of this leakage model. In particular, the Only Computation Leaks Information model [13,20] complies with our leakage model.

In the last few years a tremendous progress has been made in the interplay between provable security and side-channel attacks, such as the works [14,12,6,8,18,5] bear witness for the case of digital signatures. Admittedly, the schemes that do not use any idealized assumption (random oracle, generic groups), are much more involved than their non-leakage counterparts, and more importantly, not yet quite efficient to be used in practice. A rough estimation of the efficiency of current leakage-resilient schemes is that they are a linear number of times in the security parameter slower than their non-leakage counterparts. In this work we aim at building an efficient signature scheme secure against continual leakage. To this aim, we use an idealized model of computation called *generic bilinear group* (GBG) model, which has been previously used by Kiltz and Pietrzak [15] to provide leakage-resilient public key encryption. They propose a bilinear version of the ElGamal key encapsulation mechanism which enjoys provable leakage-resilience in the presence of continual leakage. Their scheme is very efficient, less than a handful of times slower than standard ElGamal.

We use the techniques by Kiltz and Pietrzak to propose a leakage-resilient signature scheme that builds upon the Boneh-Boyen identity-based encryption scheme [2]. The resulting signature scheme is nearly as efficient as the original identity-based encryption scheme (only $\frac{4}{3}$ times slower). Our main theorem (Theorem 2) states that allowing λ bits of leakage at every round decreases the security of the scheme by at most a factor $2^{2\lambda}$.

The main criticism that can be addressed to our work is the use of the generic group idealization to reason about side-channel attacks. The main question is whether the generic group model is a risky abstraction when side-channel attacks are considered. The main advantage of our chosen

approach lies on its practicality: the schemes obtained are of efficiency comparable to traditional schemes, a major argument in our opinion to motivate the cryptographic engineering community's interest. This alone justifies in our view a careful consideration of this approach to reason about leakage-resilient schemes, since this level of practicality is still out of reach for the existing leakage-resilient schemes in the standard model. We would like to mention that nevertheless, given the breakthroughs achieved in the last few years in the theory of leakage-resilient cryptography, we are confident that the above-mentioned efficiency gap will be progressively shrunk in the years to come and under widely accepted assumptions.

2 Definitions

In this section, we recollect some basic notions of security of signature schemes, bilinear groups, and the generic bilinear group model. We also describe the model of leakage we shall consider in this paper and formulate a definition of security of signature schemes in the presence of continual leakage. We adapt the leakage model specified in [15] to signature schemes.

Let $\mathbb{Z}$ denote the set of integers and $\mathbb{Z}_p$ $(p > 0)$ denote, depending upon the context, either the set of integers $\{0, 1, \ldots, p-1\}$ or the ring modulo p. We denote a random sampling of an element $a \in A$ from a set A, and also denote a (possibly probabilistic) output of an algorithm A, by $a \leftarrow A$. If we want to explicitly denote the randomness r used during the sampling/output, then we do so by $s \xleftarrow{r} S$. Unless otherwise mentioned or implicit from the context, any sampling is from an uniform distribution. The symbol ":=" is used to define a notation in an expression, as in $A := \mathbb{Z}$, or to explicitly indicate an output of a deterministic algorithm or a function.

2.1 Existential Unforgeability

A signature scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ consists of three probabilistic polynomial-time algorithms KeyGen, Sign, and Verify. Let κ denote the security parameter. $\mathsf{KeyGen}(\kappa)$ on input κ produces a public- and secret-key pair (pk, sk) along with other public parameters $\mathbb{PP}$. The algorithm $\mathsf{Sign}(sk, m)$ on input a secret key sk and a message $m \in M$, where M is the message space, outputs a signature σ. $\mathsf{Verify}(pk, m, \sigma)$ on input a public key pk, a message $m \in M$ and a signature σ, outputs a bit $b = 1$ meaning *valid*, or $b = 0$ meaning *invalid*. We require the following *correctness* requirement to be satisfied by Π:

$$\Pr[\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m)) = 1 \ : \ (pk, sk) \leftarrow \mathsf{KeyGen}(\kappa), m \in M] = 1.$$

The security of a signature scheme Π is defined through the following experiment:

Sign-Forge$_{\Pi}(\mathcal{A}, \kappa)$	Sign-Oracle $\Omega_{sk}(m)$
$(pk, sk) \leftarrow \mathsf{KeyGen}(\kappa)$	$w := w \cup m$
$w := \emptyset$	$\sigma \leftarrow \mathsf{Sign}(sk, m)$
$(m, \sigma) \leftarrow \mathcal{A}^{\Omega_{sk}(\cdot)}(pk)$	Return σ
If $m \in w$, then return $b := 0$	
$b \leftarrow \mathsf{Verify}(pk, m, \sigma)$	

Definition 1. [Existential Unforgeability] *A signature scheme Π is* existentially unforgeable under adaptive chosen-message attacks*, in short "secure", if $\Pr[b = 1]$ is negligible in the Experiment* Sign-Forge$_{\Pi}(\mathcal{A}, \kappa)$ *for any efficient adversary $\mathcal{A}$.*

2.2 Leakage Model

We split the secret state into two parts that reside in different parts of the memory, and structure any computation that involves access to the secret state into a sequence of steps. Any step accesses only one part of the secret state (*active* part) and the other part (*passive* part) is assumed not to leak in the current step of computation. In the case of signature schemes, we structure the signing process into two steps. For simplicity, we define a security notion for leakage-resilient signature schemes assuming that the signing process is carried out in two steps. We also refer to a single invocation of the signature generation algorithm as a *round*.

Let us consider the problem of achieving leakage resilience under continual leakage even when a significant fraction of the bits of the secret state are leaked per round. Then it is necessary that the secret state must be *stateful*, i.e. the secret state must be refreshed during every round [15]. Otherwise, after many rounds the entire secret state will be completely leaked.

Formally, a stateful signature scheme $\Pi^* = (\mathsf{KeyGen}^*, \mathsf{Sign}_1^*, \mathsf{Sign}_2^*, \mathsf{Verify}^*)$ consists of four probabilistic polynomial-time algorithms KeyGen^*, Sign_1^*, Sign_2^* and Verify^*. $\mathsf{KeyGen}^*(\kappa)$ is same as the set-up phase KeyGen of Π except that instead of a "single" secret key sk, it outputs two initial secret states (S_0, S_0'). Intuitively, S_0 and S_0' may be viewed as two shares of the secret key sk. From the point of view of an adversary, the signing algorithm Sign of Π and $(\mathsf{Sign}_1^*, \mathsf{Sign}_2^*)$ have the same functionality. First, Sign_1^* is executed and later Sign_2^* is executed. That is, the i^{th} execution of the signing process (or i^{th} round) is carried out as:

$$(S_i, w_i) \xleftarrow{r_i} \mathsf{Sign}_1^*(S_{i-1}, m_i)\ ;\ (S'_i, \sigma_i) \xleftarrow{r'_i} \mathsf{Sign}_2^*(S'_{i-1}, w_i). \tag{1}$$

In the above expression, r_i and r'_i are the randomness used by Sign_1^* and Sign_2^*, respectively. The parameter w_i is some state information passed onto Sign_2^* by Sign_1^*. The signature σ_i is generated for the message m_i, and the internal state is updated from (S_{i-1}, S'_{i-1}) to (S_i, S'_i).

We model the leakage during signature generation by giving an adversary $\mathcal{A}$ access to a leakage oracle $\Omega^{\text{leak}}_{(S_{i-1}, S'_{i-1})}(\cdot)$. This oracle, in addition to giving $\mathcal{A}$ signatures for the messages of its choice, also allows $\mathcal{A}$ to obtain leakage from the computation used to generate signatures. More precisely, let λ be a *leakage parameter*. During the i^{th} signing round, $\mathcal{A}$ is allowed to specify two functions f_i and h_i, each of range $\{0,1\}^\lambda$, that can be efficiently computed. The outputs of the leakage functions are

$$\Lambda_i = f_i(S_{i-1}, r_i)\ ;\ \Lambda'_i = h_i(S'_{i-1}, r'_i, w_i). \tag{2}$$

Since the value of m can be included in the description of f_i and h_i, hence it is not explicitly included as an input. Note that it also possible for $\mathcal{A}$ to specify h_i after obtaining Λ_i. But, for the simplicity of the exposition, we focus on the case where f_i and h_i are specified along with the message m_i to the oracle. The security of the signature scheme Π^* in the presence of (continual) leakage is defined through the following experiment Sign-Forge-Leak$_{\Pi^*}(\mathcal{A}, \kappa, \lambda)$. In the description below, $|f_i|$ refers to the length of the output of f_i.

Sign-Forge-Leak$_{\Pi^*}(\mathcal{A}, \kappa, \lambda)$	Sign-Leak-Oracle $\Omega^{\text{leak}}_{(S_{i-1}, S'_{i-1})}(m_i, f_i, h_i)$
$(pk, (S_0, S'_0)) \leftarrow \mathsf{KeyGen}^*(\kappa)$	If $\lvert f_i\rvert \neq \lambda$ or $\lvert h_i\rvert \neq \lambda$, return $\perp$
$i := 1, w := \emptyset$	$(S_i, w_i) \xleftarrow{r_i} \mathsf{Sign}_1^*(S_{i-1}, m_i)$
$(m, \sigma) \leftarrow \mathcal{A}^{\Omega^{\text{leak}}_{(S_{i-1}, S'_{i-1})}(\cdot)}(pk)$	$(S'_i, \sigma_i) \xleftarrow{r'_i} \mathsf{Sign}_2^*(S'_{i-1}, w_i)$
If $m \in w$, then return $b := 0$	$\Lambda_i := f_i(S_{i-1}, r_i)$
$b \leftarrow \mathsf{Verify}^*(pk, m, \sigma)$	$\Lambda'_i := h_i(S'_{i-1}, r'_i, w_i)$
	$i := i + 1$
	$w := w \cup m_i$
	Return $(\sigma_i, \Lambda_i, \Lambda'_i)$

Definition 2. [Existential Unforgeability with Leakage] *A signature scheme Π^* is* existentially unforgeable under adaptive chosen-message attacks in the presence of (continual) leakage *if* $\Pr[b = 1]$ *is negligible in the Experiment* Sign-Forge-Leak$_{\Pi^*}(\mathcal{A}, \kappa, \lambda)$ *for any efficient adversary $\mathcal{A}$.*

2.3 Bilinear Groups

Let $\mathsf{BGen}(\kappa)$ be a probabilistic bilinear group generator that outputs $(\mathbb{G}, \mathbb{G}_T, p, e, g)$ such that:

1. $\mathbb{G} = \langle g \rangle$ and $\mathbb{G}_T$ are (multiplicatively written) cyclic groups of prime order p with binary operations $\cdot$ and $\star$, respectively. The size of p is κ bits.
2. $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map that is:
   (a) bilinear: $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
   (b) non-degenerate: $e(g, g) \neq 1$.

Such a group $\mathbb{G}$ is said to be a bilinear group the above properties hold. It is also required that the group operations in $\mathbb{G}$ and $\mathbb{G}_T$, and the map e are efficiently computable. The group $\mathbb{G}$ is called as *base group* and $\mathbb{G}_T$ as *target group*.

2.4 Generic Bilinear Group Model

The generic bilinear group (GBG) model [4] is an extension of the generic group model [23]. The encodings of the elements of $\mathbb{G}$ and $\mathbb{G}_T$ are given by random injective maps $\xi : \mathbb{Z}_p \to \Xi$ and $\xi_T : \mathbb{Z}_p \to \Xi_T$, respectively, where Ξ and Ξ_T are sets of bit-strings. The group operations in $\mathbb{G}$ and $\mathbb{G}_T$, and evaluation of the bilinear map e are performed by three public oracles $\mathcal{O}$, $\mathcal{O}_T$ and $\mathcal{O}_e$, respectively, defined as follows. For all $a, b \in \mathbb{Z}_p$

- $\mathcal{O}(\xi(a), \xi(b)) := \xi(a + b \bmod p)$
- $\mathcal{O}_T(\xi_T(a), \xi_T(b)) := \xi_T(a + b \bmod p)$
- $\mathcal{O}_e(\xi(a), \xi(b)) := \xi_T(ab \bmod p)$

We assume that $\Xi \cap \Xi_T = \phi$, the (fixed) generator g of $\mathbb{G}$ satisfies $g = \xi(1)$, and also the (fixed) generator g_T of $\mathbb{G}_T$ satisfies $g_T = e(g, g) = \xi_T(1)$.

2.5 Min-Entropy

Let X be a finite random variable with a probability distribution Pr. The *min-entropy* of X, denoted $\mathbf{H}_\infty(X)$, is defined as $\mathbf{H}_\infty(X) := -\log_2 \left(\max_x \Pr[X = x] \right)$. Min-entropy is a standard measure of the worst-case predictability of a random variable. Let Z be a random variable. The *average conditional min-entropy* of X given Z, denoted $\tilde{\mathbf{H}}_\infty(X \mid Z)$, is defined as

$$\tilde{\mathbf{H}}_\infty(X \mid Z) := -\log_2 \left(\mathop{\mathbb{E}}_{z \leftarrow Z} \left[\max_x \Pr[X = x \mid Z = z] \right] \right).$$

Average conditional min-entropy is a measure of the worst-case predictability of a random variable given a correlated random variable. The following result is due to [9].

Lemma 1. *Let $f : X \to \{0,1\}^{\lambda'}$ be a function on X. Then $\tilde{\mathbf{H}}_\infty(X \mid f(X)) \geq \mathbf{H}_\infty(X) - \lambda'$.*

The following result is a simple variant of the Schwartz-Zippel Lemma [22,24].

Lemma 2. [Schwartz-Zippel; min-entropy version] *Let $F \in \mathbb{Z}_p[X_1, \ldots, X_n]$ be a non-zero polynomial of (total) degree at most d. Let P_i $(i = 1, \ldots, n)$ be probability distributions on $\mathbb{Z}_p$ such that $\mathbf{H}_\infty(P_i) \geq \log p - \lambda'$, where $0 \leq \lambda' \leq \log p$. If $x_i \stackrel{P_i}{\leftarrow} \mathbb{Z}_p$ $(i = 1, \ldots, n)$ are chosen independently, then $Pr[F(x_1, \ldots, x_n) = 0] \leq 2^{\lambda'} \frac{d}{p}$.*

Proof. We prove the result by induction. When $n = 1$, the univariate polynomial F has at most d roots. Since $\mathbf{H}_\infty(P_1) \geq \log p - \lambda'$, we have $\Pr[F(x_1) = 0] \leq d\, 2^{-(\log p - \lambda')} = \frac{d}{p}\, 2^{\lambda'}$.

Let us now prove the result for the n-variables case assuming the result for the $(n-1)$-variables case. On writing F as a polynomial in X_1 with coefficients in $\mathbb{Z}_p[X_2, \ldots, X_n]$, let i $(i \geq 1)$ be the degree of X_1 in the leading term and $F' \in \mathbb{Z}_p[X_2, \ldots, X_n]$ be the leading coefficient. The probability

$$\begin{aligned}\Pr[F(x_1, \ldots, x_n) = 0] \leq \Pr[F(x_1, \ldots, x_n) = 0 \mid F'(x_2, \ldots, x_n) \neq 0] \\ + \Pr[F'(x_2, \ldots, x_n) = 0].\end{aligned}$$

F' is now a non-zero polynomial, of degree at most $d - i$, in only $n - 1$ variables. By induction hypothesis we have $\Pr[F'(x_2, \ldots, x_n) = 0] \leq \frac{d-i}{p}\, 2^{\lambda'}$. When $F'(x_2, \ldots, x_n) \neq 0$, we have $\Pr[F(x_1, \ldots, x_n) = 0] \leq \frac{i}{p}\, 2^{\lambda'}$ because degree of F in X_1 is i $(i \geq 1)$ and the distributions P_i $(i = 1, \ldots, n)$ are independent. Hence $\Pr[F(x_1, \ldots, x_n) = 0] \leq \frac{d}{p}\, 2^{\lambda'}$. Note that the parameter n does not appear in the above bound. □

Corollary 1. *If $\lambda' = (1 - \epsilon) \log p$ (for constant $\epsilon > 0$) in Lemma 2, then $Pr[F(x_1, \ldots, x_n) = 0]$ is negligible (in $\log p$).*

3 Basic Signature Scheme

We now describe a signature scheme that is obtained from the Boneh-Boyen identity based encryption scheme (BB-IBE) [2]. This scheme is not

yet known to be existentially unforgeable under adaptive chosen-message attacks (EUF-CMA) in the standard model. However, we are able to prove that the BB-signature scheme is EUF-CMA secure in the GBG model.

Let $\Pi_{\mathsf{BB}} = (\mathsf{KeyGen}_{\mathsf{BB}}, \mathsf{Sign}_{\mathsf{BB}}, \mathsf{Verify}_{\mathsf{BB}})$ be a signature scheme on the message space $\mathbb{Z}_p$ defined as follows:

1. $\mathsf{KeyGen}_{\mathsf{BB}}(\kappa)$: Compute $\mathbb{PP} := (\mathbb{G}, \mathbb{G}_T, p, e, g) \leftarrow \mathsf{BGen}(\kappa)$. Choose random $x, x_0, x_1 \leftarrow \mathbb{Z}_p$. Set $X := g^x$, $X_0 := g^{x_0}$, $X_1 := g^{x_1}$ and $X_T := e(g, X) = e(g, g)^x$. The public key is $pk := (\mathbb{PP}, X_0, X_1, X_T)$ and the secret key is $sk := X$.
2. $\mathsf{Sign}_{\mathsf{BB}}(sk, m)$: Choose a random $t \leftarrow \mathbb{Z}_p$. Set $\sigma := (sk \cdot (X_0 \cdot X_1^m)^t, g^t)$. Output the signature σ.
3. $\mathsf{Verify}_{\mathsf{BB}}(pk, m, \sigma)$: Let $\sigma = (\sigma_1, \sigma_2) \in \mathbb{G}^2$. Output the bit $b = 1$ (*valid*) if $X_T \star e(\sigma_2, X_0 \cdot X_1^m) = e(\sigma_1, g)$. Otherwise output $b = 0$ (*invalid*).

Theorem 1. *The signature scheme* Π_{BB} *is* EUF-CMA *secure in the generic bilinear group model.*

Proof. Let $\mathcal{A}$ be a q-query adversary that can break the security of Π_{BB}. By a q-query adversary we mean that $\mathcal{A}$ can make totally at most q group oracle and signing oracle queries. Let $q_{\mathcal{O}}$ be the total number of calls to the group oracles $\mathcal{O}$, $\mathcal{O}_T$ and $\mathcal{O}_e$, and q_Ω correspond to the number of calls to the signing oracle. We have $q_{\mathcal{O}} + q_\Omega \leq q$. As is typical for proofs in the generic group model, we bound the advantage of $\mathcal{A}$ against Π_{BB} by the success probability of $\mathcal{A}$ in the following game $\mathcal{G}$ (see [23,19,3]). $\mathcal{A}$ plays the game $\mathcal{G}$ with an algorithm $\mathcal{B}$.

Game $\mathcal{G}$: Let X, X_0, X_1, $\{T_i : 1 \leq i \leq q_\Omega\}$, $\{U_i : 1 \leq i \leq q_g,\ 0 \leq q_g \leq 2(q_{\mathcal{O}} + 1)\}$ and $\{V_i : 1 \leq i \leq q_{g_T},\ 0 \leq q_{g_T} \leq 2q_{\mathcal{O}}\}$ be indeterminates, and $\{m_i : 1 \leq i \leq q_\Omega\}$ be elements of $\mathbb{Z}_p$ chosen by $\mathcal{A}$. Intuitively, these indeterminates correspond to randomly chosen group elements in Π_{BB}, or more precisely their discrete logarithms. The indeterminates X, X_0, X_1 correspond to the quantities x, x_0, x_1, respectively. Note that $\mathcal{A}$ might query the group oracles with representations (bit-strings) not previously obtained from the group oracles. In order to accommodate this case we introduce the indeterminates U_i, V_i. The U_i $(1 \leq i \leq q_g)$ correspond to the elements of $\mathbb{G}$, whereas V_i $(1 \leq i \leq q_{g_T})$ correspond to the elements of $\mathbb{G}_T$. We denote the lists $\{T_i : 1 \leq i \leq q_\Omega\}$, $\{U_i : 1 \leq i \leq q_g\}$ and $\{V_i : 1 \leq i \leq q_{g_T}\}$ by $\{T\}$, $\{U\}$ and $\{V\}$, respectively.

$\mathcal{B}$ maintains two lists of pairs

$$\mathcal{L} = \{(F_{1,i}\,,\, \xi_{1,i}) : 1 \leq i \leq \tau_1\}, \tag{3}$$

$$\mathcal{L}_T = \{(F_{T,i}\,,\, \xi_{T,i}) : 1 \leq i \leq \tau_T\}, \tag{4}$$

such that, at step τ ($0 \leq \tau \leq q_{\mathcal{O}}$) in the game,

$$\tau_1 + \tau_T = \tau + 2q_\Omega + q_g + q_{g_T} + 4. \tag{5}$$

The entries $F_{1,i} \in \mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}]$, $F_{T,i} \in \mathbb{Z}_p[X, X_0, X_1, \{U\}, \{V\}, \{T\}]$ are multivariate polynomials over $\mathbb{Z}_p$, whereas $\xi_{1,i}, \xi_{T,i}$ are bit-strings in the encoding sets Ξ (of $\mathbb{G}$) and Ξ_T (of $\mathbb{G}_T$), respectively. Intuitively, the polynomials in lists $\mathcal{L}$ and $\mathcal{L}_T$ correspond to elements of $\mathbb{G}$ and $\mathbb{G}_T$, respectively, that $\mathcal{A}$ will ever be able to compute or guess. In order to simplify the description, we view $\mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}]$ as a subring of $\mathbb{Z}_p[X, X_0, X_1, \{U\}, \{V\}, \{T\}]$.

Initially, $\tau = 0, \tau_1 = 2q_\Omega + q_g + 3, \tau_T = q_{g_T} + 1$,

$$\begin{aligned}
\mathcal{L} = \{\ & (1, \xi_{1,1}),\ (X_0, \xi_{1,2}),\ (X_1, \xi_{1,3}),\ \{(U_i, \xi_{1,i+3}) : 1 \leq i \leq q_g\}, \\
& \{(X + (X_0 + m_i X_1)T_i,\ \xi_{1,2i+q_g+2}),\ (T_i, \xi_{1,2i+q_g+3}) : 1 \leq i \leq q_\Omega\}\ \}, \\
\mathcal{L}_T = \{\ & X,\ \{(V_i, \xi_{T,i+1}) : 1 \leq i \leq q_{g_T}\}\ \}.
\end{aligned}$$

The bit-strings $\xi_{1,i}, \xi_{T,i}$ are set to random distinct strings from Ξ and Ξ_T, respectively. We assume that there is some ordering (say, lexicographic ordering) among the strings in the sets Ξ and Ξ_T, so that given a string $\xi_{1,i}$ or $\xi_{T,i}$, it is possible to determine its index in the lists, if it exits.

The initial state of the two lists correspond to the group elements that $\mathcal{A}$ gets as input as part of the public parameters and the signatures obtained by $\mathcal{A}$ on the messages m_i of its choice. As previously mentioned, the polynomials U_i, V_i correspond to the group elements that $\mathcal{A}$ will guess in the actual interaction. Since $\mathcal{A}$ can query the group oracles with at most two new (guessed) elements and since it may also output at most two new elements from $\mathbb{G}$ as its forgery, we have $q_g + q_{g_T} \leq 2q_{\mathcal{O}} + 2$. Hence (5) can be simplified as (assuming $q_\Omega \geq 6$, without loss of generality)

$$\tau_1 + \tau_T \leq q_{\mathcal{O}} + 2q_\Omega + 2q_{\mathcal{O}} + 2 + 4 \leq 3(q_{\mathcal{O}} + q_\Omega) \leq 3q. \tag{6}$$

The game begins by $\mathcal{B}$ providing $\mathcal{A}$ with the initial τ_1 strings $\xi_{1,1}, \ldots, \xi_{1,\tau_1}$ from $\mathcal{L}$, and τ_T strings $\xi_{T,1}, \ldots, \xi_{T,\tau_T}$ from $\mathcal{L}_T$.

Group operation: The calls made by $\mathcal{A}$ to the group oracles $\mathcal{O}$ and $\mathcal{O}_T$ are modeled as follows. For group operations in $\mathbb{G}$, $\mathcal{A}$ provides $\mathcal{B}$ with two operands (bit-strings) $\xi_{1,i}, \xi_{1,j}$ ($1 \leq i, j \leq \tau_1$) in $\mathcal{L}$ and also specifies whether to multiply or divide them. $\mathcal{B}$ answers the query by first incrementing the counters $\tau_1 := \tau_1 + 1$ and $\tau := \tau + 1$, and provides $\mathcal{A}$ with the polynomial $F_{1,\tau_1} := F_{1,i} \pm F_{1,j}$. If $F_{1,\tau_1} = F_{1,k}$ for some $k < \tau_1$, then $\mathcal{B}$ sets $\xi_{1,\tau_1} := \xi_{1,k}$. Otherwise, ξ_{1,τ_1} is set to a random string distinct

from those already present in $\mathcal{L}$. Also the pair $(F_{1,\tau_1}, \xi_{1,\tau_1})$ is appended to $\mathcal{L}$. Note that the (total) degree of the polynomials $F_{1,i}$ in $\mathcal{L}$ is at most two. Similarly, group operations in $\mathbb{G}_T$ are answered, appropriately updating the list $\mathcal{L}_T$ and the counters τ_T and τ.

Pairing: For a pairing operation, $\mathcal{A}$ queries $\mathcal{B}$ with two operands $\xi_{1,i}, \xi_{1,j}$ $(1 \leq i, j \leq \tau_1)$ in $\mathcal{L}$. $\mathcal{B}$ first increments $\tau_T := \tau_T + 1$ and $\tau := \tau + 1$, and then computes the polynomial $F_{T,\tau_T} := F_{1,i} \cdot F_{1,j}$. Again, if $F_{T,\tau_1} = F_{T,k}$ for some $k < \tau_T$, then $\mathcal{B}$ sets $\xi_{T,\tau_T} := \xi_{T,k}$. Otherwise, ξ_{T,τ_T} is set to a random string distinct from those already present in $\mathcal{L}_T$. Also the pair $(F_{T,\tau_T}, \xi_{T,\tau_T})$ is appended to $\mathcal{L}_T$. The degree of the polynomials $F_{T,i}$ in $\mathcal{L}_T$ is at most four.

When $\mathcal{A}$ terminates it outputs $(m, (\xi_{1,\alpha_1}, \xi_{1,\alpha_2})) \in \mathbb{Z}_p \times \mathcal{L} \times \mathcal{L}$ $(1 \leq \alpha_1, \alpha_2 \leq \tau_1)$. This corresponds to the "forgery" output by $\mathcal{A}$ in the actual interaction. Let the polynomials corresponding to ξ_{1,α_1} and ξ_{1,α_2} in $\mathcal{L}$ be F_{1,α_1} and F_{1,α_2}, respectively. After $\mathcal{A}$ terminates, $\mathcal{B}$ computes the polynomial

$$F_{1,\sigma} := X + F_{1,\alpha_2}(X_0 + mX_1) - F_{1,\alpha_1}. \tag{7}$$

Note that the degree of $F_{1,\sigma}$ is at most three. Next, $\mathcal{B}$ chooses random values x, x_0, x_1, $\{u\}$, $\{v\}$, $\{t\} \leftarrow \mathbb{Z}_p$ for the indeterminates X, X_0, X_1, $\{U\}$, $\{V\}$, $\{T\}$, respectively. Then it evaluates the polynomials in lists $\mathcal{L}$ and $\mathcal{L}_T$. $\mathcal{A}$ is said to have won the game $\mathcal{G}$ if:

1. $F_{1,i}(x, x_0, x_1, \{u\}, \{t\}) = F_{1,j}(x, x_0, x_1, \{u\}, \{t\})$ in $\mathbb{Z}_p$, for some two polynomials $F_{1,i} \neq F_{1,j}$ in $\mathcal{L}$.
2. $F_{T,i}(x, x_0, x_1, \{u\}, \{v\}, \{t\}) = F_{T,j}(x, x_0, x_1, \{u\}, \{v\}, \{t\})$ in $\mathbb{Z}_p$, for some two polynomials $F_{T,i} \neq F_{T,j}$ in $\mathcal{L}_T$.
3. $F_{1,\sigma}(x, x_0, x_1, \{u\}, \{t\}) = 0$ in $\mathbb{Z}_p$, and $m \neq m_i$ $\forall i$, $i = 1, \ldots, q_\Omega$.

This completes the description of the game $\mathcal{G}$.

We claim that the success probability of $\mathcal{A}$ in the actual EUF-CMA game is bounded above by its success probability in the above game $\mathcal{G}$. This is because of the following reasons:

– The conditions 1 and 2 above ensure that $\mathcal{A}$ will get to see only distinct group elements in the actual interaction. In other words, $\mathcal{A}$ is unable to cause *collisions* among group elements. As long as these two conditions are not satisfied, then the view of $\mathcal{A}$ is identical in the game $\mathcal{G}$ and the actual interaction. Hence if $\mathcal{A}$ is unable to provoke collisions, then adaptive strategies are no more powerful than non-adaptive ones (for more details, we refer to [19, Lemma 2 on pp. 12], also [23]). This observation allows us to choose group elements and their representations independently of the strategy of $\mathcal{A}$. Hence $\mathcal{A}$ specified the

messages m_i at the beginning of the game $\mathcal{G}$ and also obtained the corresponding signatures. For the same reason, it also decided at the beginning itself on the representations it would guess. Note that the assumption that $\mathcal{A}$ would a priori decide the representations it would guess is only to simplify the description of the proof and it is not an inherent limitation.
- The condition 3 above ensures that the pair $(\xi_{1,\alpha_1}, \xi_{1,\alpha_2})$ is a valid forgery on a distinct message m.

We now compute the success probability of $\mathcal{A}$ in the game $\mathcal{G}$. The τ_1 polynomials $F_{1,i}$ in $\mathcal{L}$ have degree at most two. Note that $F_{1,i} \neq F_{1,j} \Leftrightarrow F_{1,i} - F_{1,j} \neq 0$ as polynomials. From Lemma 2 (with $\lambda' = 0$), the probability that two distinct polynomials in $\mathcal{L}$ evaluate to the same value for randomly and independently chosen values for the indeterminates is at most $\frac{2}{p}$. Summing up over at most $\binom{\tau_1}{2}$ distinct pairs (i, j), the probability that the condition 1 above holds is at most $\binom{\tau_1}{2} \cdot \frac{2}{p}$. Similarly, we have the probability that the condition 2 above holds is at most $\binom{\tau_2}{2} \cdot \frac{4}{p}$. The degree of the polynomial $F_{1,\sigma}$ in condition 3 is at most three. In order to apply Lemma 2, we need to prove that F_σ is not identically equal to the zero polynomial. We prove this fact in Lemma 3 below. Let $\Pr^{forge}_{\mathcal{A},\Pi_{\mathsf{BB}}}$ denote the advantage of the adversary $\mathcal{A}$ in computing a forgery against Π_{BB}. Then, assuming Lemma 3, we obtain from (6)

$$\Pr^{forge}_{\mathcal{A},\Pi_{\mathsf{BB}}} \leq \binom{\tau_1}{2} \cdot \frac{2}{p} + \binom{\tau_2}{2} \cdot \frac{4}{p} + \frac{3}{p} \leq \frac{2}{p}(\tau_1 + \tau_2)^2 \leq \frac{18q^2}{p}. \tag{8}$$

Hence if $q = poly(\log p)$, then $\Pr^{forge}_{\mathcal{A},\Pi_{\mathsf{BB}}}$ is negligible.

Lemma 3. *The polynomial $F_{1,\sigma} \in \mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}]$ is non-zero.*

Proof. Any polynomial in $\mathcal{L}$ is obtained by either adding or subtracting two polynomials previously existing in the list. Hence we can write F_{1,α_1} and F_{1,α_2} in terms of polynomials present in $\mathcal{L}$ when it was was initialized at step $\tau = 0$ in the game $\mathcal{G}$. Note that initially $\mathcal{L}$ also includes the representations guessed by $\mathcal{A}$, in addition to the inputs.

$$\begin{aligned} F_{1,\alpha_1} = \; & c_1 + c_2 X_0 + c_3 X_1 + \textstyle\sum_{i=1}^{q_g} c_{4,i} U_i + \sum_{i=1}^{q_\Omega} c_{5,i} T_i \\ & + \textstyle\sum_{i=1}^{q_\Omega} c_{6,i}(X + (X_0 + m_i X_1) T_i), \end{aligned} \tag{9}$$

$$\begin{aligned} F_{1,\alpha_2} = \; & d_1 + d_2 X_0 + d_3 X_1 + \textstyle\sum_{i=1}^{q_g} d_{4,i} U_i + \sum_{i=1}^{q_\Omega} d_{5,i} T_i \\ & + \textstyle\sum_{i=1}^{q_\Omega} d_{6,i}(X + (X_0 + m_i X_1) T_i), \end{aligned} \tag{10}$$

where $c_j, d_j (j = 1,2,3), c_{j,i}, d_{j,i} (j = 4,5,6; 1 \leq i \leq q_\Omega) \in \mathbb{Z}_p$ are chosen by $\mathcal{A}$. We have two possible cases:

Case 1: $c_{6,i} = d_{6,i} = 0 \quad \forall i, 1 \leq i \leq q_\Omega$.

In this case, both F_{1,α_1} and F_{1,α_2} do not contain the indeterminate X. Hence the expression $F_{1,\alpha_2}(X_0 + mX_1) - F_{1,\alpha_1}$ in (7) is free of X. Therefore, in the polynomial $X + F_{1,\alpha_2}(X_0 + mX_1) - F_{1,\alpha_1}$, the coefficient of the term X is non-zero. Hence $F_{1,\sigma}$ is non-zero.

Case 2: $c_{6,k} \neq 0$ or $d_{6,k} \neq 0$ for some k, where $1 \leq k \leq q_\Omega$.

On substituting expressions from (9) and (10) into (7), we get that the coefficient of monomials $X_0^2 T_i$, $X_0 T_i$, $X_1 T_i$ in $F_{1,\sigma}$ are $d_{6,i}$, $d_{5,i} - c_{6,i}$, $m\, d_{5,i} - m_i c_{6,i}$, respectively, for $1 \leq i \leq q_\Omega$.

If $d_{6,k} \neq 0$, then the coefficient of $X_0^2 T_k$ is non-zero, and hence $F_{1,\sigma} \neq 0$. Else, $c_{6,k} \neq 0$. We again have two cases: If $d_{5,k} \neq c_{6,k}$, then the coefficient of $X_0 T_k$ is non-zero. Or else, if $d_{5,k} = c_{6,k}$, then the coefficient of $X_1 T_k$ is non-zero, since $m \neq m_i \ \forall i, i = 1, \ldots, q_\Omega$. Hence in all cases we have $F_{1,\sigma}$ to be a non-zero polynomial. □

4 A Leakage-Resilient Signature Scheme

As previously mentioned in Section 2.2, any cryptographic scheme that does not maintain a stateful secret state is insecure against continual leakage. So is the case with the signature scheme Π_{BB}. We now describe a leakage-resilient version Π^*_{BB} of Π_{BB}. We follow the techniques of [15] to adapt Π_{BB} to a leakage setting. The basic idea is to store the secret key $X = g^x$ in two different parts of the memory as $(S_0 := g^{l_0}, S'_0 := g^{x-l_0})$ for a randomly chosen $l_0 \leftarrow \mathbb{Z}_p$. Accordingly, the $\mathsf{KeyGen}_{\mathsf{BB}}$ step of Π_{BB} is modified to obtain the set-up stage $\mathsf{KeyGen}^*_{\mathsf{BB}}$ of Π^*_{BB}. The signature generation is now carried out as a two step process $\mathsf{Sign}^*_{\mathsf{BB1}}$ and $\mathsf{Sign}^*_{\mathsf{BB2}}$. During the i^{th} signature query, the two parts of the secret key (S_{i-1}, S'_{i-1}) are refreshed to obtain $(S_i := S_{i-1} \cdot g^{l_i}, S'_i := S'_{i-1} \cdot g^{-l_i})$, where $l_i \leftarrow \mathbb{Z}_p$. This is done in order to protect against continual leakage.

Let $\Pi^*_{\mathsf{BB}} = (\mathsf{KeyGen}^*_{\mathsf{BB}}, \mathsf{Sign}^*_{\mathsf{BB1}}, \mathsf{Sign}^*_{\mathsf{BB2}}, \mathsf{Verify}^*_{\mathsf{BB}})$ be a stateful signature scheme on the message space $\mathbb{Z}_p$ defined as follows:

1. $\mathsf{KeyGen}^*_{\mathsf{BB}}(\kappa)$: Compute $\mathbb{PP} := (\mathbb{G}, \mathbb{G}_T, p, e, g) \leftarrow \mathsf{BGen}(\kappa)$. Choose random $x, x_0, x_1, l_0 \leftarrow \mathbb{Z}_p$. Set $X := g^x$, $X_0 := g^{x_0}$, $X_1 := g^{x_1}$ and $X_T := e(g, X) = e(g, g)^x$. The public key is $pk := (\mathbb{PP}, X_0, X_1, X_T)$ and the secret key is $sk^* := (S_0 := g^{l_0}, S'_0 := g^{x-l_0} = X \cdot g^{-l_0}) \in \mathbb{G}^2$.
2. $\mathsf{Sign}^*_{\mathsf{BB1}}(S_{i-1}, m_i)$: Choose random $t_i, l_i \leftarrow \mathbb{Z}_p$. Set $S_i := S_{i-1} \cdot g^{l_i}$, $\sigma'_{1,i} := S_i \cdot (X_0 \cdot X_1^{m_i})^{t_i}$, and $\sigma'_{2,i} := g^{t_i}$.

3. $\mathsf{Sign}^*_{\mathsf{BB2}}(S'_{i-1}, (\sigma'_{1,i}, \sigma'_{2,i}, l_i))$: Set $S'_i := S'_{i-1} \cdot g^{-l_i}$ and $\sigma_i := (S'_i \cdot \sigma'_{1,i}, \sigma'_{2,i})$. Output the signature σ_i.
4. $\mathsf{Verify}^*_{\mathsf{BB}}(pk, m, \sigma)$: Let $\sigma = (\sigma_1, \sigma_2) \in \mathbb{G}^2$. Output the bit $b = 1$ (*valid*) if $X_T \star e(\sigma_2, X_0 \cdot X_1^m) = e(\sigma_1, g)$. Otherwise output $b = 0$ (*invalid*).

In steps 2 and 3 above, the index i keeps a count of the number of invocations (rounds) of the signing algorithm. For every $i \geq 1$, let $Y_i := \sum_{j=0}^{i} l_j$. It is easy to check that $S_i \cdot S'_i = g^{Y_i} \cdot g^{x-Y_i} = X$. We sometimes even refer to X as the secret key.

Note that $\mathsf{Sign}^*_{\mathsf{BB1}}$ requires four exponentiations and $\mathsf{Sign}^*_{\mathsf{BB2}}$ requires one. The total number of exponentiations needed for every signature invocation can be reduced from five to four if $\mathsf{Sign}^*_{\mathsf{BB1}}$ also passes on g^{l_i} to $\mathsf{Sign}^*_{\mathsf{BB2}}$. Hence only one extra exponentiation is needed when compared with the $\mathsf{Sign}_{\mathsf{BB}}$ step of Π_{BB}, which requires three.

For the sake of clarity, we would like to compare the various notations used in the signature scheme Π^*_{BB} above with those in (1) corresponding to a generic stateful signature scheme Π^*. The quantities r_i and w_i in (1) correspond to (l_i, t_i) and $(\sigma'_{1,i}, \sigma'_{2,i}, l_i)$ of Π^*_{BB}, respectively. The quantities S_i, S'_i and m_i denote the same things in both the cases. However, since the algorithm $\mathsf{Sign}^*_{\mathsf{BB2}}$ of Π^*_{BB} does not generate any randomness, there is no analogue in Π^*_{BB} for r'_i of (1). Accordingly, the leakage functions specified by an adversary to the signing oracle $\Omega^{\mathrm{leak}}_{(S_{i-1}, S'_{i-1})}(m_i, f_i, h_i)$ would be of the form $f_i(S_{i-1}, (l_i, t_i))$ and $h_i(S'_{i-1}, (\sigma'_{1,i}, \sigma'_{2,i}, l_i))$.

First we show that Π^*_{BB} is secure in the GBG model when an adversary is not allowed to obtain leakage. The following lemma is a trivial consequence of the fact that the input/output behaviour of Π^*_{BB} and Π_{BB} are identical (c.f. Theorem 1).

Lemma 4. *The signature scheme* Π^*_{BB} *is* EUF-CMA secure *in the generic bilinear group model.*

The following theorem establishes the fact that the signature scheme Π^*_{BB} is resilient to (continual) leakage attacks in the GBG model if $\lambda \ll \frac{\log p}{2}$, where λ is the leakage parameter.

Theorem 2. *The signature scheme* Π^*_{BB} *is secure with leakage w.r.t. Definition 2 in the generic bilinear group model. The advantage of a* q*-query adversary who gets at most* λ *bits of leakage per each invocation of* $\mathsf{Sign}^*_{\mathsf{BB1}}$ *or* $\mathsf{Sign}^*_{\mathsf{BB2}}$ *is* $O\left(\frac{q^2}{p} 2^{2\lambda}\right)$.

Proof. Let $\mathcal{A}$ be a q-query adversary that can break the security of Π^*_{BB}. By a q-query adversary $\mathcal{A}$ we mean that $\mathcal{A}$ can make totally at most q

group oracle and signing oracle queries. Let $q_{\mathcal{O}}$ be the total number of calls to the group oracles $\mathcal{O}$, $\mathcal{O}_T$ and $\mathcal{O}_e$, and q_Ω correspond to the number of calls to the signing oracle. We have $q_{\mathcal{O}} + q_\Omega \leq q$. In the count $q_{\mathcal{O}}$, even the group oracle queries by leakage functions f_i, h_i specified by $\mathcal{A}$ are also included.

We first informally sketch the main ideas of the proof and then formalize these ideas. Let us try to see why the proof of security of Π^*_{BB} in the absence of any leakage (i.e. proof of Theorem 1) would not carry over as it is in the presence of leakage. In the non-leakage setting, while determining the probability of collision among distinct polynomials in conditions 1-3 on page 10, we substituted for each indeterminate an independent value chosen from an uniform distribution over $\mathbb{Z}_p$. But, when $\mathcal{A}$ has access to leakage functions $f_i(S_{i-1}, (l_i, t_i))$ and $h_i(S'_{i-1}, (\sigma'_{1,i}, \sigma'_{2,i}, l_i))$, then from its point of view the parameters t_i $(1 \leq i \leq q_\Omega)$ are no longer uniformly distributed (though they are still independent). With some partial information about t_i, $\mathcal{A}$ can now cause collisions among polynomials with increased probability. Since each t_i is chosen independently and it can be leaked by only f_i, hence at most λ bits of t_i can be leaked. Apart from the values t_i, the only other "useful" information that leakage functions can provide is about the secret key $X = g^x$. This is because the parameters l_i themselves alone do not help $\mathcal{A}$ to output forgery since the signatures generated are independent of these randomly chosen values. Instead, $\mathcal{A}$ can very much use the leakages of l_i to compute, and eventually leak, the secret key X. Note that the leakage functions do not provide any additional information on the values x, x_0 or x_1.

We first bound the probability of the event that the secret key X is computed by some leakage function f_i or h_i. As long as this event has not occurred, then no bits of the secret key is leaked and the "only" additional information $\mathcal{A}$ has is about the values t_i. Clearly, the probability of this event depends on the leakage parameter λ. For instance, if the amount of leakage per invocation is not bounded, then during the first signature query itself, the adversary can leak the initial two shares of the secret key $S_0 = g^{l_0}$ and $S'_0 = X \cdot g^{-l_0}$ to recompute X. Finally, we determine the advantage of $\mathcal{A}$ conditioned on the event of the secret key X not being computed by any of the leakage functions.

Formally, we define E to be the event of computing (or guessing) the secret key $X = g^x$ by any of the leakage functions f_i or h_i $(1 \leq i \leq q_\Omega)$. Let $\overline{E}$ denote the complement of the event E, *Forgery* denote the event of $\mathcal{A}$ forging a signature on a new message, and $\Pr^{forge}_{\mathcal{A},\Pi^*_{\mathsf{BB}}} = \Pr[\text{Forgery}]$

denote the advantage of $\mathcal{A}$ in computing a forgery against Π^*_{BB}. We have

$$\Pr^{forge}_{\mathcal{A},\Pi^*_{\mathsf{BB}}} = \Pr[\text{Forgery}|E]\Pr[E] + \Pr[\text{Forgery}|\overline{E}]\Pr[\overline{E}].$$

Since $\Pr[\text{Forgery}|E]$, $\Pr[\overline{E}] \leq 1$, we obtain

$$\Pr^{forge}_{\mathcal{A},\Pi^*_{\mathsf{BB}}} \leq \Pr[E] + \Pr[\text{Forgery}|\overline{E}]. \tag{11}$$

We first bound the probability of the event E.

Lemma 5. $\Pr[E] \leq O\left(\frac{q^2}{p}2^{2\lambda}\right)$.

Proof. Let the adversary $\mathcal{A}$ play the following game $\mathcal{G}'$. Since the game $\mathcal{G}'$ is similar in nature to the game $\mathcal{G}$ in the proof of Theorem 1, we only briefly describe $\mathcal{G}'$. We use the notations introduced in the game $\mathcal{G}$. Let $\{L\}$ denote the list of indeterminates $\{L_i : 1 \leq i \leq q_\Omega\}$ that correspond to the values l_i in Π^*_{BB}.

Game $\mathcal{G}'$: For every leakage function $f_i(S_{i-1}, (l_i, t_i))$ and $h_i(S'_{i-1}, (\sigma'_{1,i}, \sigma'_{2,i}, l_i))$, $\mathcal{A}$ builds lists $\mathcal{L}^{f_i}$ and $\mathcal{L}^{h_i}$, respectively. These lists contain polynomial-bit string pairs. The polynomials are from $\mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}, \{L\}]$ and the bit-strings are from the encoding set Ξ of the group $\mathbb{G}$. Intuitively, the polynomials in lists $\mathcal{L}^{f_i}$ and $\mathcal{L}^{h_i}$ correspond to the elements of group $\mathbb{G}$ that can be computed by f_i and h_i, respectively. Every polynomial in $\mathcal{L}^{f_i}$ is of the form

$$c_{1,i}L_i + c_{2,i}\sum_{j=0}^{i-1} L_j + c_{3,i}D_i, \tag{12}$$

where $c_{1,i}$, $c_{2,i}$, $c_{3,i} \in \mathbb{Z}_p$ are chosen by $\mathcal{A}$ and $D_i \in \mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}]$ is in the list $\mathcal{L}$ (c.f. (3)). Every polynomial in $\mathcal{L}^{h_i}$ is of the form

$$d_{1,i}L_i + d_{2,i}\left(X - \sum_{j=0}^{i-1} L_j\right) + d_{3,i}\left(\left(\sum_{j=0}^{i} L_j\right) + (X_0 + m_i X_1)T_i\right) + d_{4,i}W_i, \tag{13}$$

where $d_{1,i}$, $d_{2,i}$, $d_{3,i}$, $d_{4,i}$, $m_i \in \mathbb{Z}_p$ are also chosen by $\mathcal{A}$ and $W_i \in \mathbb{Z}_p[X, X_0, X_1, \{U\}, \{T\}]$ is in the list $\mathcal{L}$.

When $\mathcal{A}$ terminates it outputs a polynomial F from the list $\mathcal{L}^{f_i}$ or $\mathcal{L}^{h_i}$, for some i. Intuitively, the polynomial F output by $\mathcal{A}$ corresponds to its guess of the secret key X. $\mathcal{A}$ is said to have won the game $\mathcal{G}'$ if

1. There is a collision in any of the lists $\mathcal{L}^{f_i}$ and $\mathcal{L}^{h_i}$, for some i ($1 \leq i \leq q_\Omega$).

2. $F - X = 0$ in $\mathbb{Z}_p$.

Note that the polynomials are now evaluated with values chosen from independent distributions with min-entropy $\log p - 2\lambda$. The reason for this will be shortly explained. This completes the description of the game $\mathcal{G}'$.

Technically speaking, $\mathcal{A}$ must also maintain lists $\mathcal{L}_T^{f_i}$ and $\mathcal{L}_T^{h_i}$ $(1 \leq i \leq q_\Omega)$ that correspond to elements of the group $\mathbb{G}_T$ that can be computed by f_i and h_i. To simplify the discussion, we only describe collisions in the lists $\mathcal{L}^{f_i}$ and $\mathcal{L}^{h_i}$. Similar arguments apply for the lists $\mathcal{L}_T^{f_i}$ and $\mathcal{L}_T^{h_i}$. Since we compute $\Pr[E]$ only up to a constant factor, the additional advantage $\mathcal{A}$ obtains from collisions in $\mathcal{L}_T^{f_i}$ and $\mathcal{L}_T^{h_i}$ is implicitly included. However, working on the lines of the proof of Theorem 1, it is relatively straightforward to completely formalize the present discussion.

For similar reasons as given in the proof of Theorem 1, we have $\Pr[E]$ is bounded above by the success probability of $\mathcal{A}$ in the above game $\mathcal{G}'$. We particularly like to note the following. As observed in [1, pp. 691] and the references therein, even in the leakage setting adaptive strategies are no more powerful than non-adaptive ones.

Before computing the success probability of $\mathcal{A}$, we first show that $F - X$ is a non-zero polynomial. From Lemma 4 and Theorem 1, we know that Π_{BB}^* is secure without leakage. Hence the polynomial X (that corresponds to the secret key) cannot appear in the list $\mathcal{L}$, because this would otherwise imply that the secret key can be computed without access to leakage functions. A formal proof for this fact can be easily obtained on the lines of the proof of Lemma 3. Hence even when $c_{1,i} = c_{2,i} = 0$ in (12), the lists $\mathcal{L}^{f_i}$ cannot contain the polynomial X. If $c_{1,i} \neq 0$ or $c_{2,i} \neq 0$, then the polynomial in (12) will contain either L_i or L_{i-1}, or both. Hence the polynomial X cannot appear in any of the lists $\mathcal{L}^{f_i}$. In a similar way it can be seen that the lists $\mathcal{L}^{h_i}$ do not contain X. Hence $F - X$ is a non-zero polynomial of degree at most two.

Let us now determine the probability that the condition 1 above holds, i.e. the probability of collisions among distinct polynomials in any of the lists $\mathcal{L}^{f_i}$ and $\mathcal{L}^{h_i}$. In order to compute the probability, we evaluate the polynomials in (12) and (13) by choosing values from $\mathbb{Z}_p$ according to (independent) distributions with min-entropy at least $\log p - 2\lambda$. This is because $\mathcal{A}$ can obtain at most 2λ bits of leakage about l_i $(i = 0, \ldots, q_\Omega)$, and at most λ bits of t_i $(i = 1, \ldots, q_\Omega)$. From Lemma 1, the values l_i, t_i have min-entropy at least $\log p - 2\lambda$ in the view of $\mathcal{A}$. The total length of the lists $\mathcal{L}^{f_i}$, $\mathcal{L}^{h_i}$ is at most $O(q_\Omega + q_\mathcal{O}) = O(q)$. Hence there can be at most $O(q^2)$ pairs of distinct polynomials (of degree at most two)

evaluating to the same value. From Lemma 2 (with $\lambda' = 2\lambda$), we obtain $\Pr[E] \leq O\left(\frac{q^2}{p}2^{2\lambda}\right)$. Since $F - X$ is a non-zero polynomial of degree at most two, the probability that $F - X$ evaluates to zero is at most $\frac{2}{p}2^{2\lambda}$. This probability is also implicitly included in the above bound. □

We now determine the probability $\Pr[\text{Forgery} \mid \overline{E}]$ in (11).

Lemma 6. $\Pr[\text{Forgery} \mid \overline{E}] \leq \dfrac{18q^2}{p}2^{\lambda}$.

Proof. Given that the event E has not occurred, the only meaningful leakage $\mathcal{A}$ can now obtain is that of t_i $(i = 1, \ldots, q_\Omega)$. Since at most λ bits of t_i can leak (only by f_i), from the view point of $\mathcal{A}$ the values t_i have min-entropy at least $\log p - \lambda$. From Lemma 2 (with $\lambda' = \lambda$), the probability of collision among distinct polynomials in conditions 1-3 on page 10 is now increased by a factor of 2^{λ}. Hence, from (8), we obtain $\Pr[\text{Forgery} \mid \overline{E}] \leq \frac{18q^2}{p}2^{\lambda}$. □

From (11) and Lemmas 5 and 6, we have $\mathrm{Pr}^{forge}_{\mathcal{A},\Pi^*_{\mathsf{BB}}} \leq O\left(\dfrac{q^2}{p}2^{2\lambda}\right)$. This completes the proof of Theorem 2. □

Acknowledgements. We like to thank Jean-Sébastien Coron for his valuable comments on an early draft of this paper.

References

1. Divesh Aggarwal and Ueli Maurer. The leakage-resilience limit of a computational problem is equal to its unpredictability entropy. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT*, volume 7073 of *LNCS*, pages 686–701. Springer, 2011.
2. Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.
3. Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.
4. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 440–456. Springer, 2005.
5. Elette Boyle, Gil Segev, and Daniel Wichs. Fully leakage-resilient signatures. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *LNCS*, pages 89–108. Springer, 2011.
6. Zvika Brakerski, Yael Tauman Kalai, Jonathan Katz, and Vinod Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS*, pages 501–510. IEEE Computer Society, 2010.

7. Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Çetin Kaya Koç and Christof Paar, editors, *CHES*, volume 1717 of *LNCS*, pages 292–302. Springer, 1999.
8. Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *LNCS*, pages 613–631. Springer, 2010.
9. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
10. Stefan Dziembowski and Sebastian Faust. Leakage-resilient cryptography from the inner-product extractor. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT*, volume 7073 of *LNCS*, pages 702–721. Springer, 2011.
11. Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.
12. Sebastian Faust, Eike Kiltz, Krzysztof Pietrzak, and Guy N. Rothblum. Leakage-resilient signatures. In Daniele Micciancio, editor, *TCC*, volume 5978 of *LNCS*, pages 343–360. Springer, 2010.
13. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *LNCS*, pages 463–481. Springer, 2003.
14. Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *LNCS*, pages 703–720. Springer, 2009.
15. Eike Kiltz and Krzysztof Pietrzak. Leakage resilient elgamal encryption. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *LNCS*, pages 595–612. Springer, 2010.
16. Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Neal Koblitz, editor, *CRYPTO*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.
17. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *LNCS*, pages 388–397. Springer, 1999.
18. Tal Malkin, Isamu Teranishi, Yevgeniy Vahlis, and Moti Yung. Signatures resilient to continual leakage on memory and computation. In Yuval Ishai, editor, *TCC*, volume 6597 of *LNCS*, pages 89–106. Springer, 2011.
19. Ueli M. Maurer. Abstract models of computation in cryptography. In Nigel P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *LNCS*, pages 1–12. Springer, 2005.
20. Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *TCC*, volume 2951 of *LNCS*, pages 278–296. Springer, 2004.
21. Krzysztof Pietrzak. A leakage-resilient mode of operation. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *LNCS*, pages 462–482. Springer, 2009.
22. Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
23. Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.
24. Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *EUROSAM*, volume 72 of *LNCS*, pages 216–226. Springer, 1979.